A.A. 2019/20
Docente: Biagio Lenzitti

UNIVERSITÀ
DEGLI STUDI
DI PALERMO

Reti di Calcolatori

RELAZIONE FINALE SUL PROGETTO "STAZIONE METEOROLOGICA"

Massimiliano Poma (N. Matricola: 0640469)

Rosario Mulè (N. Matricola: 0643221)

Andrea Ventura (N. Matricola: 0644505)

Studenti: **Massimiliano Poma**, **Rosario Mulè**, **Andrea Ventura**

Sommario

1. Introduzione

Lo scopo del nostro progetto è quello di realizzare, mediante l'utilizzo di 2 sensori:

1. **DHT11**
2. **BMP180**

e tramite l'uso del NodeMCU integrato con una scheda **WiFi ESP8266**, una **stazione meteorologica in miniatura** che sia in grado di rilevare:

1. La **temperatura**
2. L'**umidità**
3. La **pressione**

dell'ambiente circostante in cui questa è posizionata. Questa stazione meteorologica poi deve essere in grado di inviare i dati rilevati in un database e tali dati, infine, dovranno essere mostrati in una pagina web. Analizzeremo, adesso, i componenti utilizzati per la realizzazione del nostro progetto; successivamente, invece, porremo la nostra attenzione sul codice che ci ha permesso di programmare la nostra NodeMCU, inviare dati al database e realizzare la pagina web.

1.1 ESP8266

Il componente ESP8266 è un dispositivo che è in grado di connettersi ad una rete WiFi in modo da realizzare una serie di funzioni grazie ad un'adeguata programmazione. Grazie a questo siamo stati in grado di acquisire informazioni ed inviarle sulla rete. In particolare l'uso, nel nostro caso, ci ha permesso di inviare i dati che sono stati letti dai sensori di umidità, temperatura e pressione. Un'immagine di tale pezzo è il seguente:

1.2 DHT11

Il sensore DHT11 è un sensore utilizzato per la misurazione dell'umidità e della temperatura dell'ambiente in cui esso è posizionato. Il sensore è costituito da 3 pin:

1. **VCC**: la tensione di **alimentazione positiva**;
2. **GND**: la tensione di **alimentazione negativa**;

3. **Data**: è il **segnale** lungo il quale viaggiano i dati.

Come detto sopra, grazie a questo sensore noi siamo in grado di rilevare l'umidità relativa, cioè un indice che ci permette di definire la quantità di vapore contenuta nell'aria. Nel caso in cui si ha un'umidità relativa pari:

1. Al 100%, significa che l'umidità è la massima possibile;

2. Allo 0%, significa che non vi è del vapore nell'aria, quindi l'aria sarà secca.

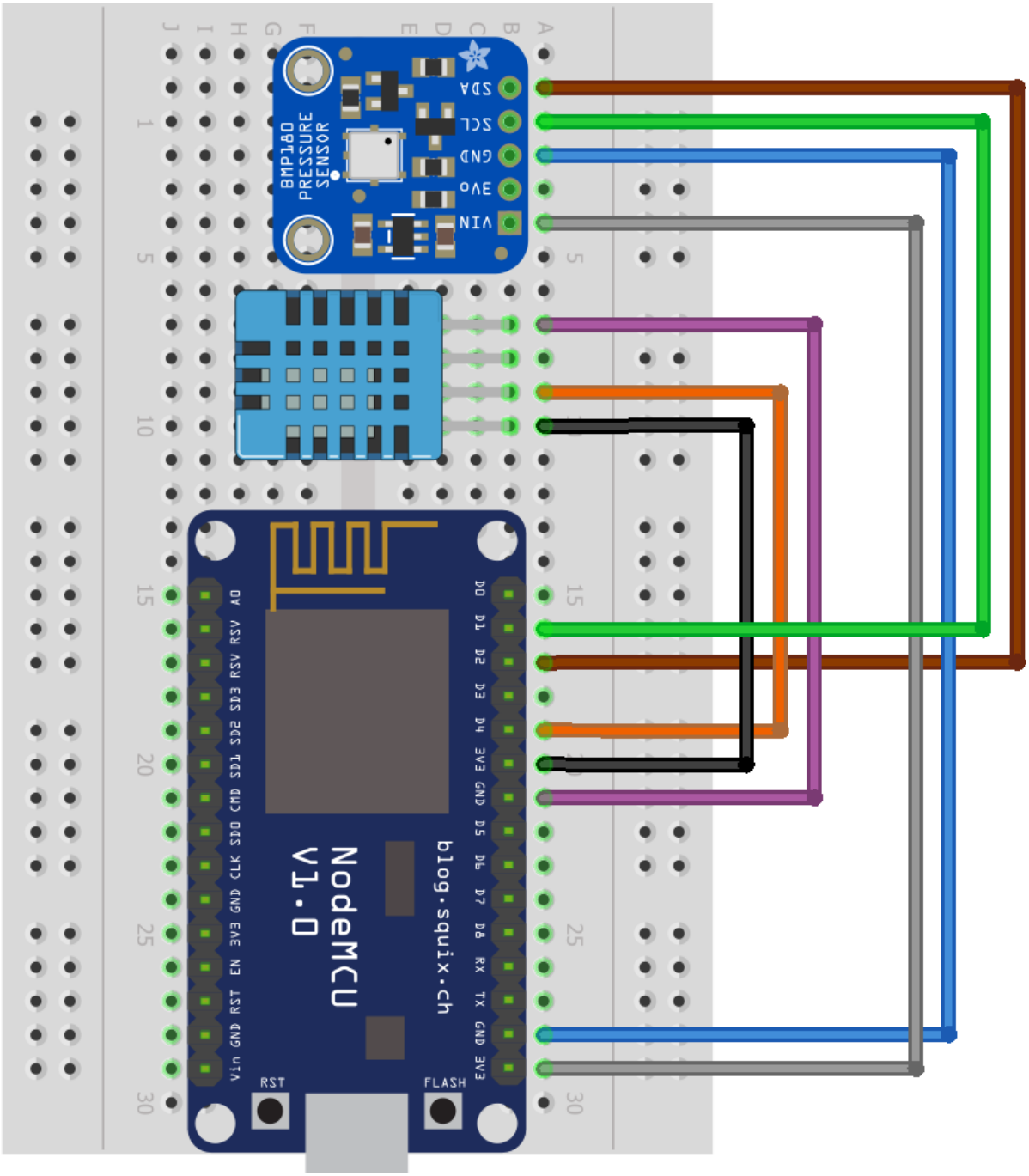
Il range di misura dell'umidità che è possibile rilevare con tale sensore va da un minimo di 20% RH[1] ad un massimo di 90% RH. L'accuratezza è pari al 4% RH, il che significa che il sensore potrebbe commettere un errore pari a tale valore sia in positivo che in negativo. Altresì, nel caso in cui si verifica una brusca variazione dell'umidità, il sensore all'incirca potrebbe impiegare 10 secondi per rilevarla. Per quanto concerne, invece, la temperatura, il sensore è in grado di rilevare solamente temperature negative e positive inferiori a 50°C. Nel caso in cui si verifica una variazione brusca di temperatura, il sensore potrebbe impiegare all'incirca 13 secondi per rilevarla. Un'immagine di questo sensore è il seguente:

1.3 BMP180

Il sensore BMP180 ci permette di misurare la temperatura, la pressione atmosferica e l'altitudine. Nel nostro caso però, tale componente viene utilizzato esclusivamente per la rilevazione della pressione atmosferica. La pressione atmosferica è la pressione causata dal peso dell'aria che preme sulla Terra. L'unità di misura nel SI per la pressione è il Pascal, indicato con Pa. Il BMP180 rileva i dati con l'unità di misura del Pascal, ma questi, successivamente vengono convertiti in hectoPascal, dove indicheremo questa unità di misura con hPa. Un'immagine di questo sensore è il seguente:

[1] RH sta ad indicare "Umidità Relativa"

2. Rappresentazione circuitale



3. Spiegazione del file arduino

Il codice contenuto nel file arduino ci permette di acquisire dei dati dai nostri sensori DHT11 e BMP180 e inviare tali dati ad un database presente su un server. Per fare ciò abbiamo utilizzato un server con installato **Apache** come server web, interprete PHP e database Mysql. Il client su arduino, dopo che ha acquisito i dati rilevati dai due sensori, andrà ad eseguire una connessione all'indirizzo IP del server sulla porta 80, richiamando una pagina PHP e inviando i dati. Successivamente il server web andrà ad eseguire la pagina

PHP, che dopo aver acquisito i dati inviati da arduino mediante il metodo GET, effettuerà una richiesta di connessione al database, al quale invierà i dati per la memorizzazione. Andiamo ad analizzare nel dettaglio il nostro codice:

1. Abbiamo importato le seguenti librerie:

```
#include <ESP8266WiFi.h>
#include <SFE_BMP180.h>
#include "dht.h"
```

   a. La libreria < ESP8266WiFi.h > ci sarà utile nel momento in cui vogliamo andare ad utilizzare delle funzioni che ci permettono di collegarci alla rete WiFi;

   b. La libreria < SFE_BMP180.h > è una libreria che abbiamo importato per utilizzare delle funzioni caratteristiche del sensore BMP180 per la rilevazione di temperatura, pressione e altitudine;

   c. La libreria < dht.h >, invece, è la libreria che abbiamo importato per utilizzare le funzioni relative al sensore DHT11.

2. Abbiamo dichiarato due costanti, in cui abbiamo inserito il nome e la password della rete WiFi a cui intendiamo connetterci. Questo viene effettuato dalle seguenti righe di codice:

```
const char *wifi_ssid = "FRITZ!Box 7490"; //Nome della rete
const char *wifi_password = "96049200348625511764"; //Password della rete
```

3. Abbiamo dichiarato tre variabili, tutte di tipo double, in cui andiamo a memorizzare, di volta in volta, i valori che sono stati rilevati dai vari sensori. In particolare, le variabili valueTemperature e valueHumidity sono dedite alla memorizzazione della temperatura e della pressione, mentre per quanto concerne la variabile valuePressure, in questa andremo a memorizzare il valore della pressione. Ciò viene effettuato dalle seguenti righe di codice:

```
double valueTemperature; //Variabile per la temperatura
double valueHumidity;  //Variabile per l'umidità
double valuePressure; //Variabile per la pressione
```

4. Altre due variabili che abbiamo dichiarato sono una costante per memorizzare l'indirizzo IP del terminale e una variabile di tipo stringa in cui, come vedremo più avanti, andremo ad inserire la stringa inviata da arduino dopo la connessione. Il codice è il seguente:

```
const char* server = "192.168.178.36"; //Indirizzo IP del terminale
String strURL = ""; //Stringa inviata da arduino dopo la connessione
```

5. Altre variabili dichiarate, che ci sono utili ai fini della rilevazione dei dati e della connessione al client sono le seguenti:

```
dht DHT;  //Memorizziamo un'istanza del DHT11
SFE_BMP180 pressure; //Memorizziamo un'istanza del BMP180
WiFiClient client;
```

6. Nella sezione "setup" abbiamo "acceso" i nostri sensori, impostato il numero di Baud con cui lavorerà l'ESP8266, e inserito una funzione che ci permetterà di connetterci alla rete WiFi . Abbiamo anche inserito un controllo per verificare se effettivamente il sensore BMP180 è collegato o meno. Le seguenti righe di codice fanno quanto detto:

```
void setup(void)
{

  Serial.begin(9600); //Numero di Baud con cui lavora l'ESP826
  Serial.println("");
  delay(1000);//Tempo necessario per l'accensione del sensore

  //Verifichiamo se il sensore BMP180 è connesso
  if(pressure.begin())
  {
    Serial.println("BMP180 connesso!");
  }
  else
  {
    Serial.println("BMP180 non connesso");
  }

  Serial.print("Connessione a ");
  Serial.println(wifi_ssid);
  setup_wifi(); //Connessione alla rete WiFi
}
```

Come detto pocanzi, in questo frammento di codice è stata inserita la funzione $setup\_wifi()$ che ci permette di connetterci alla rete WiFi. Questa funzione è stata sviluppata nel seguente modo:

```
void setup_wifi()
{
  WiFi.begin(wifi_ssid, wifi_password);
  Serial.print("Mi sto connettendo");

  while (WiFi.status() != WL_CONNECTED)
  {
    delay(500);
    Serial.print(".");
  }

  Serial.println("");
  Serial.print("Connessione riuscita! ");
}
```

La connessione effettiva al WiFi viene fatta invocando la funzione:

```
WiFi.begin(wifi_ssid, wifi_password);
```

Il processo di connessione può richiedere alcuni secondi e verificheremo se questo è stato completato nel seguente ciclo:

```
while (WiFi.status() != WL_CONNECTED)
{
  delay(500);
  Serial.print(".");
}
```

Tale ciclo continuerà ad essere ripetuto fin quando `WiFi.status()` è diverso da `WL_CONNECTED`.

7. Nella sezione "loop" andiamo a leggere inizialmente i valori della temperatura, dell'umidità e della pressione. Memorizzeremo questi nelle variabili apposite e successivamente andremo a creare l'url che ci permetterà di connetterci al server. Verranno quindi poi inviati, tramite i file PHP che spiegheremo successivamente, i valori rilevati dai sensori e tali verranno memorizzati nel database. Ogni 30 secondi invieremo una richiesta http GET e successivamente verrà rieseguita la sezione "loop". Particolare attenzione poniamo sulle funzioni che ci permettono di rilevare i dati:

    a. `startPressure()`: tale funzione invia il comando per avviare la misurazione della pressione. A questo passiamo un valore che può essere compreso tra 0 e 3. Un valore pari a 3 fornisce un'alta risoluzione, ma anche un ritardo più lungo tra le misurazioni. Un valore pari a 0 fornisce una risoluzione inferiore, ma è più veloce. Questa funzione restituisce il numero di millisecondi che occorre attendere prima di leggere il valore della pressione del sensore.

b. `getPressure()`: tale funzione ci permette di leggere il valore della pressione e memorizzarlo nella variabile `valuePressure`. A questa funzione passiamo anche la variabile `valueTemperature`, perché il calcolo della pressione dipende dalla temperatura.

Tutto ciò verrà effettato dal seguente frammento di codice:

```
void loop(void)
{
  DHT.read11(D4);
  valueTemperature = DHT.temperature; //Acquisizione della temperatura
  valueHumidity =  DHT.humidity; //Acquisizione dell' umidità

  char status; //Variabile per memorizzare il tempo necessario al sensore per compiere una misurazione
  status = pressure.startPressure(3);

  if(status != 0)
  {
    delay(status);
    status = pressure.getPressure(valuePressure,valueTemperature); //Acquisizione della pressione in hPa
  }

  if(client.connect(server,80))
  {
    //Creo L'url utilizzando una stringa
    strURL = "GET /add.php?Temperatura=";
    strURL +=  valueTemperature;
    strURL += "&Umidita=";
    strURL += valueHumidity;
    strURL += "&Pressione=";
    strURL += valuePressure;
    strURL += " HTTP/1.1";

    //Invio della richiesta al server
    client.println(strURL);
    client.println("Host: arduino");
    client.println("Connection: close");
    client.println();

    client.stop();    //Chisusura della connessione
  }

  delay(30000); //Manda una richiesta HTTP GET ogni 30 secondi*/
}
```

4. Spiegazione dei file in PHP/HTML

4.1 File Connect

Il seguente file PHP permette di collegarci al database creato sul server. In questo andremo a specificare il nome dell'host, dell'user, la password e il nome del database. Successivamente andremo a verificare se effettivamente questa connessione è andata a buon fine o meno. Tutto ciò viene effettuato dal seguente codice:

```php
<?php

    function Connection(){
        $host="localhost";
        $user="root";
        $pass="";
        $db="misurazioni";

        $connessione = new PDO("mysql:host=$host;dbname=$db",$user,$pass);

        if (!$connessione) {
            echo("Errore nella connessione");
        }

        return $connessione;
    }
?>
```

4.2 File Add

Questo file ci permette di prendere i dati rilevati dai sensori e inviati dall'ESP8266 e mediante una query SQL andremo a inserire questi dati all'interno del database da noi creato. Questo viene effettuato dalle seguenti righe di codice:

```php
<?php
    include("connect.php");

    $link=Connection();

    $temp1 = $_GET['Temperatura'];
    $hum1=$_GET['Umidita'];
    $pres1=$_GET['Pressione'];

    $query = $link->exec("INSERT INTO `stazione_metereologica` (`ID`, `Timestamp`, `Temperatura`, `Umidita`, `Pressione`)
    VALUES (NULL, current_timestamp(),'$temp1', '$hum1', '$pres1')");

    $link = null;
?>
```

4.3 File Index

Grazie a questo file andremo a leggere i dati contenuti all'interno del nostro database e mediante l'ausilio del codice HTML all'interno del file PHP andiamo a creare una pagina web in cui verrà creata una tabella e in tale tabella vi sarà un report di tutti i dati che sono stati rilevati. Questo viene effettuato dalle seguenti righe di codice:

```
<!DOCTYPE html>
    <html>
        <head>
        <meta http-equiv="refresh" content="35">
            <style>
            body{
                padding:10px;
                color:  #000000;
                font-variant: small-caps;
                font-size:5em;
                text-align: center;
            }
            h2{
                font-size:0.3em;
                text-align: center;
                padding:10px;
            }

            </style>
        </head>
            <body style="background-color: #f0f8ff">
            <header>Risultati stazione Meteo</header>
            <h2>
                <table align = "center" border=”1” cellspacing=”1” cellpadding=”1”>
                <tr>
                <td width="30%" bgcolor="#e6e6fa"> Timestamp </td>
                <td bgcolor="#90ee90"> Temperatura </td>
                <td bgcolor="#ffffff"> Umidità </td>
                <td bgcolor="#f08080"> Pressione </td>
                </tr>
                <?php

                  include("connect.php");


                   $link2=Connection();

                   $query2 = $link2 -> query("SELECT Timestamp,Temperatura,Umidita,Pressione FROM
                   `stazione_metereologica` ORDER BY Timestamp");
                   $i = 0;
                       while($row = $query2 ->fetch(PDO::FETCH_BOTH) ){
                           echo '<tr>
                                   <td bgcolor="#e6e6fa"> ' . $row['Timestamp'] . '</td>
                                   <td bgcolor="#90ee90"> ' . $row['Temperatura'] . " °C" . '</td>
                                   <td bgcolor="#ffffff"> ' . $row['Umidita'] .  " %" .'</td>
                                   <td bgcolor="#f08080"> ' . $row['Pressione'] . " hPa" . '</td>
                               </tr>';
                           $i++;
                       }

                ?>
                </table>
            </h2>
            </body>
</html>
```